Automatic Transfer Function Design via MLLM-Assisted 2D Semantic Decomposition

Haejin Jeong and Won-Ki Jeong

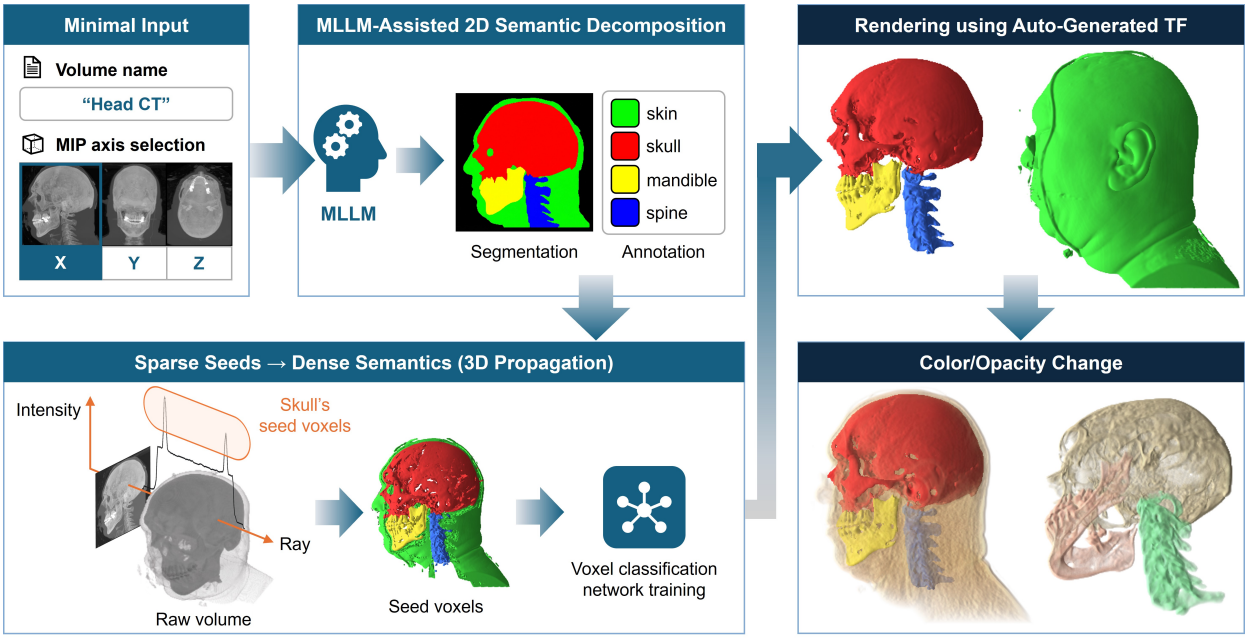


Figure 1: Overview of our framework for automatic semantic transfer function (TF) design. Given only minimal user input—a volume name and an MIP axis—we generate a single projection that is semantically decomposed by an MLLM into labeled regions. These 2D semantic cues are lifted into sparse 3D seed voxels and propagated through a voxel classification network to obtain dense voxel-level semantics. The resulting semantic TF enables intuitive, part-aware volume rendering without manual TF design. After the TF is automatically generated, users can interactively adjust the color and opacity of each semantic part.

Abstract—Designing transfer functions (TFs) for direct volume rendering is a fundamental yet labor-intensive task, as users need to carefully explore the relationship between voxel attributes and visual outcomes to reveal structures of interest. Despite advances in learning-based approaches, many existing methods depend on repeated user input. Furthermore, many methods relying on conventional data value-based TFs, which use voxel intensity or gradient information, often struggle to separate voxels into meaningful semantic groups. To address these problems, we present a framework that enables automatic TF design via multimodal large language model (MLLM)-assisted 2D semantic decomposition, requiring only minimal user input, namely the volume name and a projection axis. Starting from a single maximum intensity projection of the volume, our method transforms it into a semantic segmentation mask using an MLLM, which is then lifted into 3D as a sparse set of reliable voxel cues. To convert these sparse cues into a complete semantic representation, we introduce a voxel classification network trained with a confidence-aware propagation loss, resulting in dense voxel-level semantic assignments that are readily translated into TFs. Despite relying solely on single-image segmentation, our method achieves precise part-level separation. Evaluation on a range of volume datasets shows that ours produces better semantic renderings while significantly reducing the effort and time required for TF design compared to state-of-the-art techniques.

Index Terms—Direct volume rendering, transfer function, multimodal large language model

1 Introduction

Direct volume rendering (DVR) is a technique for exploring volumetric datasets in scientific visualization, enabling users to reveal structures within complex 3D data such as medical scans and simulation outputs. A critical step in DVR is the design of transfer functions (TFs) that map voxel attributes such as intensity to optical properties including color and opacity. Effective TFs allow users to find and highlight meaningful structures in the data. However, designing such TFs remains a challenging and time-consuming process that often requires extensive trial-and-error and domain expertise. In particular, 1D TFs [9], which assign color and opacity to voxels based on intensity, and 2D TFs [19], which additionally incorporate gradient magnitude, require users to manually identify which intensity or gradient magnitude ranges correspond to specific parts of the volume and to explicitly define boundaries between them.

To alleviate this burden, several approaches have been proposed that incorporate user input into the TF design process. For example, users may annotate points or brush strokes on volume slices [11], or select voxels from a feature space such as a t-SNE embedding derived from

• *The authors are with the Department of Computer Science and Engineering, Korea University, Seoul, Republic of Korea. E-mail: {haejinjeong, wkjeong}@korea.ac.kr.*
• *Won-Ki Jeong is a corresponding author.*

Manuscript received xx xxx. 202x; accepted xx xxx. 202x. Date of Publication xx xxx. 202x; date of current version xx xxx. 202x. For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org. Digital Object Identifier: xx.xxxx/TVCG.202x.xxxxxxx

volumetric features [37]. These methods enable users to specify regions of interest and iteratively refine the visualization. Because users can inspect the rendering results after each interaction and make further adjustments, the process allows the visualization to progressively converge toward the desired outcome. Compared to directly manipulating 1D or 2D TFs, these interaction-based approaches typically require fewer manual parameter adjustments to obtain satisfactory rendering results. However, such methods still rely on multiple rounds of user interaction to achieve the desired visualization.

More recently, several studies have explored the use of vision–language models, multimodal large language models (MLLMs), and diffusion models to design TFs using natural language prompts. Despite reducing the need for manual interaction, these approaches still exhibit notable limitations. Some methods operate by manipulating a 1D TF [14, 23], restricting the achievable results to those expressible within a 1D intensity mapping. Although there exist approaches that perform volume rendering using 3D Gaussian Splatting (3DGS) [15] and learn visual attributes at the level of 3D Gaussians [3, 32]—thereby not being restricted to intensity-based mappings as in traditional 1D TF—these methods still require an initial TF or segmentation results.

Research on separating regions of interest has been actively explored in both 3D medical image segmentation and general 3D scene segmentation. Dataset-specific supervised models [13, 38] can achieve high accuracy but require retraining for each new dataset, while recent medical segmentation foundation models [12, 24, 25] improve generality but remain limited by their medical-image training domains. In computer vision, 3D segmentation methods for meshes, point clouds, NeRF [27], and 3DGS commonly apply vision foundation models such as SAM [18] or DINO [5] to multi-view images and lift the resulting masks or features into 3D [8, 35, 36]. However, directly applying these approaches to DVR datasets is challenging because target structures are often intermingled with surrounding materials and may not be visually separable without an appropriate TF. Consequently, vision foundation models applied to raw volume renderings often fail to provide reliable semantic separation, particularly for complex structures such as thin bonsai branches or overlapping skin and skull regions in head data.

Motivated by these limitations, we leverage recent MLLMs as semantic guidance for TF design. MLLMs have shown strong image-understanding and segmentation capabilities, suggesting an opportunity to extract semantic cues from rendered volume images and guide DVR beyond purely intensity-based or manual TF design. We introduce a framework that converts MLLM-generated segmentation masks into semantic TFs for volume rendering. Given a single maximum intensity projection (MIP) image, an MLLM generates a 2D mask of semantically meaningful structures, which is projected back into the volume to obtain sparse, high-confidence voxel seeds for each semantic region. To propagate these sparse labels throughout the volume, we design a voxel classification network with a confidence-weighted propagation loss. The resulting dense voxel-level semantic labels are then used to construct semantic TFs for intuitive and meaningful volume rendering. Experiments on several volumetric datasets show that our method effectively reveals semantic structures with minimal manual effort.

The contributions of this work are summarized as follows:

- We introduce a framework that formulates TF design as a semantic lifting problem, converting MLLM-generated 2D semantic decomposition from a single MIP into a volumetric semantic representation.
- We design a voxel classification network that combines positional and attribute cues through a learnable fusion mechanism, together with a confidence-aware propagation loss that spreads sparse semantic supervision to the entire volume.
- Our method requires only minimal input—a volume name and a projection axis—and eliminates iterative user interaction, thereby significantly reducing user effort while still achieving high-quality semantic rendering.
- We validate our method on diverse volumetric datasets, demonstrating that it consistently outperforms existing approaches in semantic rendering quality while also requiring less time.

2 Related Work

2.1 Transfer Function Design

Early TF design methods primarily relied on intensity-based editing using histograms or multi-dimensional feature spaces, enabling users to identify structures based on the statistical distributions of voxel attributes. 1D TFs [9] map voxel intensity directly to color and opacity, while 2D TFs [19] incorporate additional features such as gradient magnitude to better separate material boundaries and structures. Extending these traditional TF designs with higher-dimensional voxel attributes, Two-level TF [37] proposes a two-level TF framework that integrates a conventional 2D intensity–gradient magnitude TF with a secondary TF constructed using t-SNE embedding. In this approach, voxels are first roughly selected in the 2D TF domain and then projected into a low-dimensional t-SNE space derived from multidimensional voxel attributes. The resulting embedding reveals clusters of similar voxels, enabling finer segmentation.

To further reduce the manual effort involved in TF specification, learning-based approaches have been introduced for TF design, which inspired the development of our method. Tzeng et al. [33] define a TF that maps voxel attributes to color and opacity by training a machine learning model on user-labeled samples obtained through painting on slices and using the trained model to classify voxels. Building upon this approach, Soundararajan and Schultz [31] trained supervised classification models to estimate material probabilities. To visualize classification uncertainty, they introduce probabilistic TFs that determine color and opacity through probability-based weighted averaging of predefined material colors and opacities. Kim et al. [17] employ a convolutional neural network to generate color mappings for TFs based on example images, enabling image-driven TF specification. Differentiable Design Galleries [28] learns a continuous latent design space of TFs using neural rendering and deep generative models. In this framework, rendered images are encoded into latent representations corresponding to implicit TFs, allowing users to query, navigate, and modify TF designs in the image domain before reconstructing explicit TFs through differentiable volume rendering.

Another line of work leverages learned feature representations and large pretrained models to facilitate TF specification and expressive volume visualization. Engel et al. [11] propose a segmentation-based TF design approach that utilizes semantic features extracted from a self-supervised Vision Transformer (ViT). Instead of training a model for each dataset, their method employs a frozen pretrained ViT to compute feature embeddings for volumetric data and identifies semantically similar structures based on feature similarity to user annotations, enabling interactive annotation-guided TF design without additional model training. T2TF [14] introduces a text-guided TF design approach that leverages vision–language models to generate semantic TFs from textual descriptions. In this framework, candidate TFs are evaluated based on the similarity between rendered images and user-provided text prompts, enabling semantic TF generation without manual parameter tuning. TexGS-VolVis [32] further incorporates multiple large pretrained models to support expressive scene editing in volume visualization. Built on a textured Gaussian splatting representation that decouples geometry and appearance, the system integrates InstructPix2Pix [4] for text-driven scene editing, CLIP [29] for image-driven non-photorealistic editing, and SAM for 2D segmentation, which is lifted to the 3D Gaussian representation to enable localized editing of volumetric structures.

Beyond TF design itself, recent work explores integrating large language models into visualization systems to support natural language interaction. ParaView-MCP [23] introduces an autonomous visualization agent that connects an MLLM with the ParaView visualization platform [2] through a structured tool interface. By enabling the language model to issue commands to ParaView's API and observe visualization outputs, the system allows users to interact with complex visualization workflows—including TF editing—through conversational instructions and iterative visual feedback. IntuiTF [34] formulates TF design as an iterative Trial-Insight-Replanning cycle and implements it through an evolution-driven explorer paired with an MLLM-guided evaluator. The

explorer encodes TFs as Gaussian mixture genomes and performs mutation, crossover, and selection to generate candidate solutions, while the evaluator employs an MLLM to perform pairwise comparisons between renderings, assessing both quality criteria and alignment with user-specified text or image intent. In these two approaches, the resulting TFs remain a 1D mapping defined solely over intensity values; voxels belonging to different anatomical or material structures but sharing similar intensity ranges cannot be disambiguated, which is precisely the limitation that conventional data value-based TFs face.

Despite these advances, many existing approaches still rely on manual TF editing, repeated user interaction, handcrafted feature engineering, or a 1D TF defined solely over intensity values. In contrast, our approach constructs semantic TFs by leveraging MLLM-generated segmentation and propagating sparse semantic seeds throughout the volume, reducing the need for manual TF tuning, repeated user interaction, and the reliance on a 1D intensity-based TF.

2.2 3D Segmentation

In DVR, TFs are commonly used to separate semantic parts for visualization, making TF design closely related to 3D segmentation. Similar to interactive TF design, numerous interactive approaches have been proposed for 3D segmentation. For example, Slice-and-Conquer [7] proposes a planar-to-3D framework that constructs accurate 3D masks from a small number of annotated 2D slices by using the resulting 2D mask predictions as shape priors, thereby reducing both user effort and computational cost. SegVol [10] introduces a foundation model for volumetric segmentation using unified spatial and semantic prompts. These approaches demonstrate that, when user prompts are carefully designed and effectively utilized, accurate and efficient segmentation results can be achieved.

Beyond prompt-based interaction, recent advances in segmentation have explored leveraging vision-language models to reduce annotation dependency. VCLIPSeg [21] introduces a CLIP-enhanced medical image segmentation model that integrates text embeddings at the voxel level to capture semantic relationships. SATR [1] addresses zero-shot 3D shape segmentation by transferring knowledge from 2D vision-language models, inferring 3D semantic regions from multi-view 2D predictions. These approaches highlight the growing trend of incorporating cross-modal semantics.

Building upon this trend toward foundation models, SAM and its variants, vision foundation models with impressive zero-shot segmentation performance, have been widely adopted for 3D segmentation. Gaussian Grouping [36] introduces identity encoding for each 3D Gaussian by using SAM predictions. GARField [16] optimizes a scale-conditioned affinity field within radiance fields from 2D masks by SAM, allowing hierarchical grouping of scene elements across multiple levels of granularity. SAMPart3D [35] achieves scalable zero-shot 3D part segmentation and distills SAM for multi-granularity decomposition. PartField [22] learns a continuous 3D feature field via distillation from 2D masks by SAM2 [30], producing consistent and hierarchical part segmentation. These approaches demonstrate the effectiveness of distilling knowledge from 2D foundation models into 3D representations.

SAM-based segmentation approaches have also been extended to 3D medical imaging. MedSAM [24] adapts SAM into the medical domain by fine-tuning it on a large-scale dataset across diverse modalities, enabling robust and generalizable segmentation performance beyond task-specific models. VISTA3D [12] introduces a unified 3D segmentation foundation model that supports both automatic and interactive segmentation within a single framework. It addresses that fine-tuning SAM on 2D medical data has limitations in adapting to 3D images, and instead distills SAM by constructing 3D supervoxels from 2D SAM feature maps. SAM3D [6] enables zero-shot semi-automatic segmentation in 3D medical images by combining sparse user prompts, multi-axis slicing, and 2D SAM inference to reconstruct 3D masks without additional training.

Despite these advances, many existing methods still rely on spatial prompts such as points or bounding boxes provided by users. Moreover, 3D scalar field datasets used in DVR often contain complex and overlapping structures, making it difficult to achieve precise part decomposition using vision foundation models such as SAM or DINO without manually defining an initial TF. To address this limitation, we leverage MLLMs, which provide more refined 2D semantic decomposition than existing vision foundation models, applied to an MIP that captures the overall structure of the volume without requiring an initial TF, enabling automatic TF generation.

3 Our Approach

Our framework automatically constructs a semantic TF from minimal user input by combining MLLM-assisted semantic decomposition with a learning-based volumetric propagation scheme. Given only a volume name and a projection axis, our method produces a dense semantic representation that can be directly used for volume rendering, without requiring iterative manual interaction.

Figure 1 shows an overview of our framework. The pipeline begins by generating an MIP image and performing MLLM-assisted 2D semantic decomposition to obtain labeled regions with representative colors, providing a global semantic view. These regions are then lifted into 3D by selecting reliable seed voxels based on intensity peaks and assigning confidence scores. A voxel classification network propagates semantic information from these sparse seeds to the full volume using a dual-branch design with confidence-weighted and neighborhood-based losses. Finally, a semantic TF is constructed from voxel-wise semantic probabilities, enabling intuitive control of opacity and appearance while preserving standard volume rendering.

3.1 MLLM-Assisted 2D Semantic Decomposition

Our goal is to automatically generate a high-quality semantic TF without relying on any initial TF. While slice-based inspection is a common approach, it is difficult to perceive complete structures, as individual parts may appear fragmented or partially occluded across slices. To address this limitation, we adopt a MIP as a global representation that captures the overall structure and individual components of the volume in a single image.

Given a volumetric dataset, we generate a 2D MIP image $\mathbf{I} \in \mathbb{R}^{H \times W}$ along a user-specified axis. Among the three canonical axes (x, y, and z), the user selects the projection direction that best reveals the overall structure of the volume while minimizing overlap between semantic parts. We then perform semantic decomposition of the projection image using an MLLM in two stages.

In the first stage, we prompt the MLLM to generate a colored segmentation mask $\mathbf{S}$ from the MIP while strictly preserving the original geometry. The prompt enforces several constraints: (i) the segmentation must remain strictly within the original silhouette without introducing new structures, (ii) fine structures and gaps between parts must be preserved without merging or dilation, and (iii) boundaries must remain sharp with no color bleeding across regions. Rather than requiring the user to manually specify the number of target parts, we instruct the MLLM to autonomously identify and distinguish between 3 to 5 semantically meaningful parts. This range is deliberately constrained: without such an instruction, the MLLM tends to produce overly simplistic segmentations that separate only the foreground from the background; conversely, allowing too many parts often causes the model to generate separate masks for regions that are not clearly distinguishable in the MIP. In addition, the MLLM is instructed to assign each part a color from a predefined discrete palette $C = \{\texttt{red}, \texttt{blue}, \texttt{yellow}, \texttt{green}, \texttt{cyan}, \texttt{magenta}, \texttt{purple}\}$. As a result, each pixel in $\mathbf{S} \in C^{H \times W}$ is assigned exactly one color corresponding to a semantic part.

In the second stage, the colored segmentation mask $\mathbf{S}$, together with the volume name, is fed back to the MLLM to obtain semantic annotations $A = \{(l_i, c_i)\}_{i=1}^{K}$. Here, l_i denotes a semantic text label (e.g., *leaves*, *branches*, *pot*), and $c_i \in C$ is the corresponding color used in $\mathbf{S}$. The prompt enforces structured JSON output and restricts each part to a single unambiguous name, avoiding composite labels. It also ensures strict consistency between the colors used in the segmentation mask and those reported in the annotations.

In both stages, we use predefined prompts in which only the user-provided volume name is automatically substituted before being fed

to the MLLM. To ensure that the method automatically operates effectively regardless of the given dataset, the prompts do not include any dataset-specific structural descriptions.

Finally, the semantic decomposition is defined as $R = (\mathbf{S}, A)$. This decomposition step provides a high-level semantic interpretation of the volume from a single projection, effectively transforming low-level intensity variations into structured semantic regions. Notably, this process requires only the volume name and a MIP axis as input, enabling fast and convenient extraction of semantic information without additional user interaction.

3.2 Seed Voxel Construction

In MIP, only the maximum-intensity voxel along each pixel ray is represented, which makes it inappropriate to uniformly assign the segmentation mask to all voxels along the ray. To lift the semantic decomposition into the 3D voxel space, we first extract a set of reliable seed voxels from the segmentation mask and assign a confidence score to each seed voxel to quantify its reliability.

Ray Construction Each pixel (u,v) in the segmentation mask $\mathbf{S}$ is associated with a semantic part index $k(u,v) \in \{1,\dots,K\}$, which corresponds to a semantic label $l_{k(u,v)}$ defined in A. Each pixel defines a projection ray along the selected axis. For each pixel (u,v), we extract the intensity profile along its corresponding ray: $\mathbf{r}_{u,v} = \{I(z,u,v) \mid z = 1,\dots,D\}$, where z denotes the depth index along the projection direction, and D is the number of voxels along the ray.

Intensity Peak-Based Voxel Selection. In MIP, the pixel intensity is dominated by the highest-intensity structures along the ray. These structures typically correspond to visually salient or semantically meaningful components of the volume. Therefore, we perform intensity peak-based voxel selection to identify the most representative voxels along each ray. We first identify candidate voxels using a high-intensity mask:

$$\mathbf{M}_{u,v}(z) = \mathbb{I}\big(I(z,u,v) \geq \tau_{\min} \cdot I^{u,v}_{\max}\big) \tag{1}$$

where $I^{u,v}_{\max} = \max_z I(z,u,v)$ denotes the maximum intensity along the ray, and $\tau_{\min} \in (0,1)$ is a threshold that controls the minimum relative intensity required for a voxel to be considered as a candidate with respect to the maximum intensity along the ray.

However, high-intensity regions may still include noise or elongated structures that are not representative of distinct semantic parts. To refine the selection, we focus on contiguous regions that contain prominent peaks along the ray.

$$\mathbf{P}_{u,v}(z) = \mathbb{I}\big(I(z,u,v) - I^{u,v}_{\min} \geq \tau_{\text{prom}} \cdot I^{u,v}_{\max}\big) \tag{2}$$

where $I^{u,v}_{\min} = \min_z I(z,u,v)$ denotes the minimum intensity along the ray, and $\tau_{\text{prom}} \in (0,1)$ controls the required prominence of a voxel, ensuring that only sufficiently salient peaks relative to the intensity range along the ray are selected.

We then define the set of selected voxel indices along the ray as $Z_{u,v} = \{z \mid \mathbf{M}_{u,v}(z) = 1 \wedge \mathbf{P}_{u,v}(z) = 1\}$. This selection ensures that only voxels corresponding to strong and reliable peaks along the ray are retained.

Boundary-Aware Spatial Weighting. Pixels near semantic boundaries are more likely to be ambiguous, as small segmentation errors can lead to incorrect semantic assignments. To reduce the influence of such unreliable pixels, we introduce a spatial weighting scheme based on the distance to region boundaries. We define the boundary pixel set as $B = \{(u,v) \mid \exists (u',v') \in N_4(u,v)$ such that $k(u',v') \neq k(u,v)\}$, where $N_4(u,v)$ denotes the 4-neighborhood of pixel (u,v). For each pixel (u,v), we compute the distance to the nearest boundary as $d(u,v) = \min_{(u',v')\in B} \sqrt{(u-u')^2 + (v-v')^2}$. Based on this distance, we define a spatial weight:

$$s_{u,v} = 1 - \exp\left(-\frac{d(u,v)}{\tau_{\text{dist}}}\right) \tag{3}$$

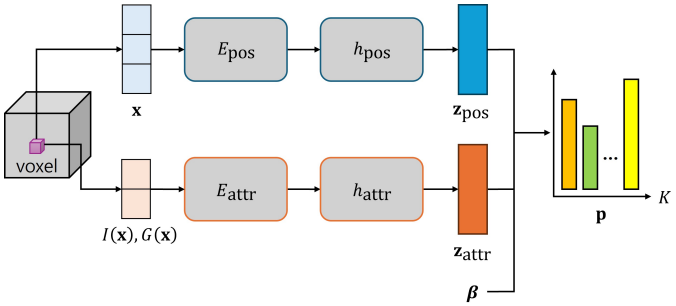


Figure 2: Architecture of our voxel classification network. A dual-branch design encodes spatial coordinates and voxel attributes (intensity and gradient magnitude), which are adaptively fused via a learnable gating vector β and normalized with softmax to predict semantic part probabilities.

This formulation assigns low weights to pixels near boundaries and gradually increases the weight toward the interior of semantic regions, thereby suppressing unreliable contributions. Here, $\tau_{\text{dist}} > 0$ is a distance scale parameter governing this rate of increase. We adopt this exponential saturation form because it provides a smooth, monotonically increasing weight that is bounded in $(0,1)$, avoiding the abrupt transitions of a hard distance threshold while still concentrating the suppression effect near semantic boundaries, where segmentation errors are most likely to occur.

Confidence Assignment. For each selected depth index $z \in Z_{u,v}$, we assign a confidence score:

$$w_{u,v}(z) = \frac{I(z,u,v)}{I^{u,v}_{\max}} \tag{4}$$

This indicates how close the voxel is to the maximum-intensity voxel along the same ray. We then combine this with the spatial weighting from Eq. 3 to obtain the final confidence:

$$\hat{w}_{u,v}(z) = w_{u,v}(z) \cdot s_{u,v}, \quad z \in Z_{u,v} \tag{5}$$

Thus, the confidence score reflects how confident the voxel is with respect to the semantic part associated with it by the segmentation mask.

Seed Voxel Labeling. Each selected voxel (z,u,v) with $z \in Z_{u,v}$ inherits the semantic part index $k(u,v)$. The output of this stage is a sparse set of seed voxels $V_{\text{seed}} = \{(\mathbf{x}_i, k_i, w_i)\}$, where $\mathbf{x}_i = (z,u,v)$, $k_i = k(u,v)$, and $w_i = \hat{w}_{u,v}(z)$. Here, $k_i \in \{1,\dots,K\}$ serves as the semantic label of the seed voxel.

3.3 Voxel Semantic Classification

While the seed voxels provide reliable semantic cues, they are inherently sparse and do not cover the entire volume. To construct a dense semantic TF, we train a voxel classification network that propagates semantic information from the seed voxels to the rest of the volume.

Voxel-Wise Semantic Prediction. We define a voxel classification network $f_\theta(\mathbf{x}, I(\mathbf{x}), G(\mathbf{x})) : \mathbb{R}^5 \to \mathbb{R}^K$ that predicts semantic logits for any voxel (Fig. 2). Given a spatial coordinate $\mathbf{x} \in [0,1]^3$, we sample its corresponding intensity $I(\mathbf{x})$ and gradient magnitude $G(\mathbf{x})$. To effectively combine the geometric and attribute-based evidence, we adopt a dual-branch architecture. This design is motivated by the observation that different semantic parts are best distinguished by different cues. The positional branch takes the per-voxel spatial coordinate and captures geometric context, enabling the network to distinguish parts that are spatially separated but share similar intensity profiles—such as the spine and the mandible in the head dataset. In contrast, the attribute branch takes the per-voxel intensity and gradient magnitude as input, allowing the network to distinguish parts that differ in intensity or gradient characteristics—such as the skin and the skull, which are spatially adjacent yet exhibit substantially different intensity distributions. A

single shared representation may fail to capture such heterogeneous discriminative factors.

In the positional branch, the spatial coordinate is encoded using a multi-resolution hash grid encoder E_{pos} to obtain a positional feature $\mathbf{f}_{\text{pos}}$, which captures high-frequency geometric structures. In the attribute branch, the sampled intensity and gradient magnitude are concatenated and passed through an attribute encoder E_{attr} (a lightweight MLP) to produce an attribute feature $\mathbf{f}_{\text{attr}}$. This transformation maps low-dimensional scalar inputs into a higher-dimensional feature space and alleviates scale and distribution differences between geometric and attribute inputs, enabling stable joint learning.

The positional and attribute features are processed independently through two heads to produce branch-specific logits, $\mathbf{z}_{\text{pos}} = h_{\text{pos}}(\mathbf{f}_{\text{pos}})$ and $\mathbf{z}_{\text{attr}} = h_{\text{attr}}(\mathbf{f}_{\text{attr}})$, where h_{pos} and h_{attr} are MLPs with a single hidden layer and ReLU activation. The fused logits $\mathbf{z}$ are obtained by a part-wise linear combination:

$$\mathbf{z} = \beta \odot \mathbf{z}_{\text{pos}} + (1-\beta) \odot \mathbf{z}_{\text{attr}} \tag{6}$$

where $\beta \in [0,1]^K$ is a learnable part-wise gating vector (parameterized via a sigmoid function). This gating mechanism allows the network to adaptively assign different importance to geometric and attribute cues for each semantic part, enabling more flexible and part-aware decision boundaries.

The logits are converted into a probability distribution over semantic parts using the softmax function, $\mathbf{p}_i = \text{softmax}(\mathbf{z}_i)$, where $\mathbf{p}_i \in \mathbb{R}^K$ represents the predicted semantic part probability of voxel i.

Neighborhood-Based Propagation. The propagation mechanism is designed to extend reliable semantic supervision from sparse seed voxels to the full volume through neighborhood aggregation embedded in the loss formulation. We exploit local consistency using 6-neighborhood connectivity $N(\mathbf{x}_i)$. Each voxel is represented using intensity and gradient magnitude as $\mathbf{f}_i = [I(\mathbf{x}_i), G(\mathbf{x}_i)]$. The similarity between voxel i and its neighbors j is computed as $d_{i,j} = \|\mathbf{f}_i - \mathbf{f}_j\|^2$. Based on this distance, we compute the affinity weight:

$$a_{i,j} = \frac{\exp\left(-\frac{d_{i,j}}{2\sigma^2}\right)}{\sum_{j' \in N(\mathbf{x}_i)} \exp\left(-\frac{d_{i,j'}}{2\sigma^2}\right)} \tag{7}$$

For each voxel, the affinity weights quantify the likelihood that a neighboring voxel belongs to the same semantic part, based on the similarity of their intensity and gradient magnitude. Using the weights, we construct a neighborhood-based soft target distribution:

$$\mathbf{q}_i = \sum_{j \in N(\mathbf{x}_i)} a_{i,j}\, \mathbf{p}_j \tag{8}$$

where $\mathbf{q}_i \in \mathbb{R}^K$ aggregates the semantic predictions of neighboring voxels and serves as a propagated supervisory signal.

At the same time, we propagate confidence across the volume, where propagation is applied only to non-seed voxels while seed voxels retain their original confidence:

$$\tilde{w}_i = \begin{cases} w_i, & \text{if } i \text{ is a seed voxel} \\ \eta \sum_{j \in N(\mathbf{x}_i)} a_{i,j}\, w_j, & \text{otherwise} \end{cases} \tag{9}$$

where $\eta \in (0,1]$ is an attenuation factor that prevents over-confident accumulation during propagation.

Confidence-Aware Propagation Loss. We enforce consistency between the prediction at voxel i and the neighborhood-derived distribution using KL divergence:

$$\mathscr{L}_{\text{prop}} = \sum_i (1-\tilde{w}_i)\, \text{KL}\big(\mathbf{q}_i \,\|\, \mathbf{p}_i\big) \tag{10}$$

This formulation encourages stronger propagation in uncertain regions while preserving stable predictions in high-confidence areas.

Supervised Loss on Seed Voxels. For seed voxels, we use the semantic part index k_i from the segmentation mask $\mathbf{S}$ as the ground-truth label and apply a confidence-weighted cross-entropy loss:

$$\mathscr{L}_{\text{sup}} = \sum_{i \in V_{\text{seed}}} w_i \cdot \big(-\log p_i^{(k_i)}\big) \tag{11}$$

where $p_i^{(k_i)}$ denotes the predicted probability of the ground-truth part k_i.

Final Training Loss. The final training loss is defined as:

$$\mathscr{L} = \mathscr{L}_{\text{sup}} + \lambda_{\text{prop}} \mathscr{L}_{\text{prop}} \tag{12}$$

This learning strategy enables effective propagation of sparse semantic cues throughout the volume, resulting in a dense and semantically consistent TF defined in the following section.

3.4 Semantic Volume Rendering

Rendering Formulation. After training the voxel classification network, we obtain a dense semantic probability field over the volume. Based on this, we define TF for the i-th sample $\mathbf{x}_i$ as follows:

$$\mathbf{TF}(\mathbf{x}_i) = (\mathbf{c}_i, \alpha_i), \quad \mathbf{c}_i = C_{k_i^*}, \quad \alpha_i = 1 - \exp(-\sigma_i \delta_i) \tag{13}$$

where $\mathbf{c}_i$, α_i, σ_i, and δ_i, denote the color, opacity, density, and step size at the i-th sample, respectively. k_i^* is the predicted semantic part index at the i-th sample defined as follows:

$$k_i^* = \arg\max_k p_i^{(k)} \tag{14}$$

where $p_i^{(k)}$ is the predicted probability of the part k at the i-th sample location. Using k_i^*, we assign the corresponding color $\mathbf{c}_i = C_{k_i^*}$ from the predefined color palette C. Density-based opacity follows the standard volume rendering formulation derived from the absorption-emission model [26].

To make opacity reflect voxel intensity, we define the density to be proportional to the voxel intensity. Given a sampling location i, the base density is defined as:

$$\sigma_i = \lambda_{k_i^*} I_i \tag{15}$$

where I_i denotes the interpolated voxel intensity at the i-th sample location, and $\lambda_{k_i^*} > 0$ is the density scaling parameter for the k_i^* part. The accumulated color $\mathbf{C}$ along each ray is then computed as:

$$\mathbf{C} = \sum_i T_i\, \alpha_i\, \mathbf{c}_i, \quad T_i = \prod_{j<i} (1-\alpha_j). \tag{16}$$

Note that users can interactively modify the color and opacity of each part by changing part colors defined in the color palette C and modifying the density scaling factor λ_k, respectively.

Semantic-Aware Density Modulation. To smoothly suppress background regions, we modulate the density using the predicted probability of the background part. Let $p_i^{(\text{bg})}$ denote the predicted probability of the background part, where the background is included as one of the K semantic parts. We define:

$$\sigma_i' = \sigma_i \cdot \big(1 - p_i^{(\text{bg})}\big) \tag{17}$$

This formulation reduces opacity near the boundary between foreground parts and background, enabling smooth transitions.

To further remove background regions, we apply an intensity-based threshold:

$$\sigma_i'' = \sigma_i' \cdot \mathbb{I}(I_i > \tau_{\text{int}}) \tag{18}$$

The threshold τ_{int} is automatically determined from the segmentation mask $\mathbf{S}$. Specifically, we collect intensity values of pixels classified as background in $\mathbf{S}$, remove outliers, and set τ_{int} as the maximum remaining intensity.

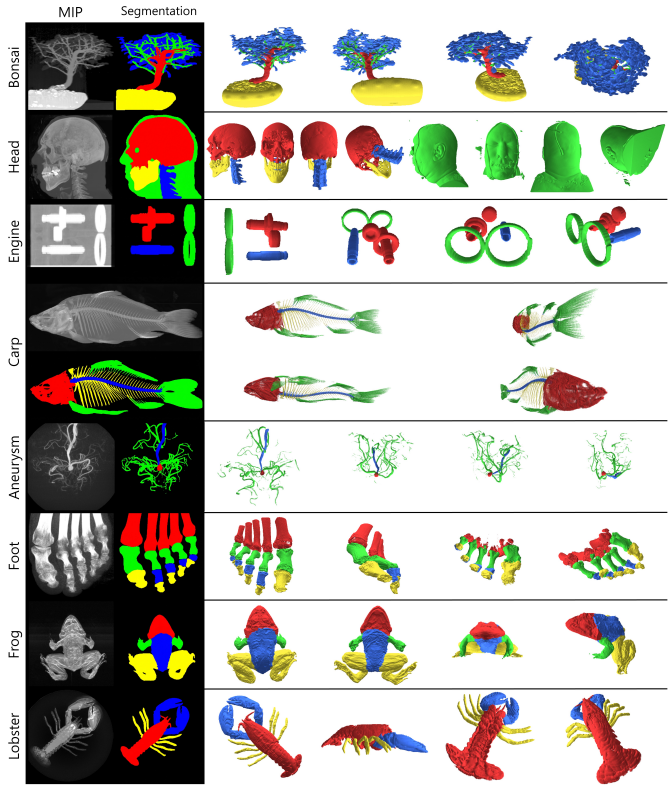


Figure 3: Our qualitative results across multiple datasets. From left to right, each row shows the input MIP, the MLLM-generated 2D segmentation, and the corresponding semantic rendering results from different viewpoints.

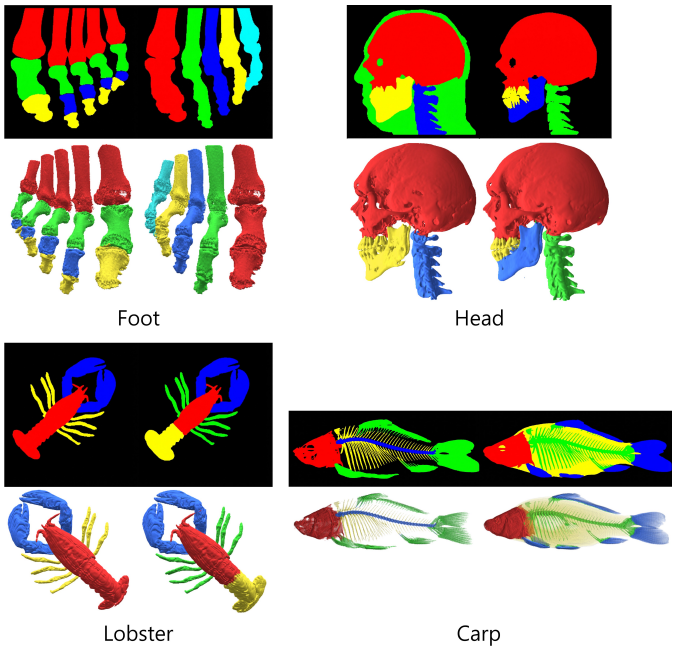


Figure 4: MLLM-generated segmentation masks and corresponding semantic volume rendering results for the foot, head, lobster, and carp datasets. For each dataset, two different semantic decompositions are presented side-by-side.

4 Results

4.1 Implementation Details

For generating the colored segmentation mask $\mathbf{S}$, we use the MLLM *Gemini 3 Pro Image Preview (Nano Banana Pro)*. For generating semantic annotations A, we use *Gemini 3 Pro Preview*. We implement our network in PyTorch and train it for 500 iterations using the Adam optimizer with a base learning rate of $\mathrm{lr} = 1 \times 10^{-3}$. Different components of the network are assigned different learning rates: the multi-resolution hash grid encoder uses $0.3 \times \mathrm{lr}$, the attribute encoder uses $0.5 \times \mathrm{lr}$, both positional and attribute heads use lr, and the part-wise gating vector β uses $2 \times \mathrm{lr}$. The other parameters used in our method are set as follows: $\tau_{\mathrm{min}} = 0.95$, $\tau_{\mathrm{prom}} = 0.1$, $\tau_{\mathrm{dist}} = 0.5$, $\sigma = 0.1$, $\eta = 0.1$, $\lambda = 5000$, and $\lambda_{\mathrm{prop}} = 70$. To more realistically reveal object surfaces, we apply a normal-based shading model inspired by classical gradient-based volume shading methods [20], which approximate surface appearance using local gradients. We use a system equipped with an Intel Xeon Gold 6326 CPU, 1TB RAM, and an NVIDIA RTX A6000 GPU with 48 GB memory. All results presented in this section are obtained using the settings described above.

4.2 Qualitative Results

Figure 3 presents semantic volume rendering results automatically generated by our method. By leveraging the MLLM-assisted 2D semantic decomposition obtained from a single MIP, our method successfully partitions and visualizes the entire volume into semantic parts across all eight experimental datasets using the same parameter settings.

In the bonsai dataset, for the given viewpoint, leaves located in front of the branches and trunk— which are not visible in the MIP due to their relatively low intensity—are nevertheless correctly classified. Similarly, in the head dataset, the large skin region overlapping with the skull, mandible, and spine is also not visible in the MIP. Despite this, our method successfully visualizes the skin as a coherent structure that appropriately encloses the other parts. In the engine dataset, our method also successfully separates internal empty regions within target parts, even though these regions are not distinguished in the segmentation mask.

In the carp dataset, it accurately isolates the thin and numerous rib structures. In the aneurysm dataset, it effectively separates fine, overlapping vessels. In the foot, frog, and lobster datasets, our method likewise successfully separates parts that cannot be distinguished based on intensity alone, using only a single image.

Figure 4 presents MLLM-generated segmentation masks and the corresponding semantic volume rendering results for the foot, head, lobster, and carp datasets. For each dataset, two different semantic decompositions are shown in the left and right columns. This demonstrates that the MLLM can produce diverse semantic decompositions even for the same input, and that our method can successfully reflect these variations.

In the foot dataset, the two columns differ in their classification criteria. The left column follows an anatomical structure-based decomposition, where parts are categorized according to bone types, such as metatarsals, proximal phalanges, middle phalanges, and distal phalanges. In contrast, the right column classifies each toe (e.g., thumb, index, middle, ring, and pinky) as an individual part. In the head dataset, the right column further separates the teeth compared to the left column. In the lobster dataset, the tail is additionally separated in the right column. In the carp dataset, the body is further separated in the right column.

4.3 Comparison with Related Work

We compare our method against the following related works. Specifically, we examine whether each method can produce semantic volume rendering results comparable to ours, and how effectively each approach performs part separation under the given experimental datasets. The resolution, size, and the number of target parts for each dataset used in the comparison are reported in Tab. 1.

4.3.1 Compared Methods and Experimental Setup.

ParaView-MCP [23] connects an MLLM with ParaView to enable volume rendering through natural language prompts by iteratively updating a 1D TF. We utilized the text prompts provided in the paper and the accompanying video as much as possible, and additionally incorporated prompts for part separation to obtain the results. We generated results using ParaView-MCP in two settings: first, by using only text prompts

to instruct the model to visualize the target parts with specified target colors (hereafter referred to as *text-only*), and second, by providing the MIP and the colored segmentation mask **S** from our method as inputs to Claude and instructing it to perform part separation and coloring to match the given mask (hereafter referred to as *2D semantic*). Given a user prompt, the system iteratively updates the TF until the objective is satisfied; we waited until Claude stopped further updates to obtain the final result. While the original paper used Claude's Sonnet 3.7 as the MLLM, we instead employed a more recent and higher-performance version, Sonnet 4.6.

SAM3D [6] extends SAM to enable zero-shot semi-automatic 3D medical image segmentation. Given user-provided positive and negative point prompts for a target part, polylines are constructed and used as guidance. The volume is then sliced along multiple axes, and SAM is applied to each slice using polyline-derived prompts, followed by recomposition into a 3D segmentation. Although SAM3D was originally proposed for medical image segmentation, it is a zero-shot method not trained specifically on medical data, making it applicable to a wide range of datasets commonly used in DVR. Many SAM-based methods widely used in 3D segmentation use input formats such as meshes, point clouds, or multi-view images, which require an initial TF when applied to 3D scalar field datasets used in volume rendering. In contrast, SAM3D directly operates on volumetric data such as CT or MRI scans, making it a representative baseline among SAM-based 3D segmentation methods for our setting.

The number of annotated slices and point annotations varies depending on the size, shape, and structural complexity of each part. For each part, we provided point annotations on 3 to 10 slices, with tens of points per slice. In this work, a transform refers to a distinct slicing orientation applied to the 3D volume, which generates multiple intersecting 2D views for SAM-based segmentation and improves accuracy through redundancy. We selected the maximum number of transforms (10) available in the authors' implementation to achieve the most accurate 3D segmentation. All other parameters were kept at their default values. The final rendered images were generated using ParaView.

4.3.2 Qualitative Results Comparison

Figure 5 presents visualization results of target parts for eight datasets using our method, ParaView-MCP (text-only), ParaView-MCP (2D semantic), and SAM3D. We also include the MIP and the segmentation mask **S** used in our method and in ParaView-MCP (2D semantic). For most datasets, a consistent viewpoint is used to facilitate comparison across methods; however, when certain parts are occluded in specific results, alternative viewpoints are adopted to better reveal the target parts.

Our method successfully separates and visualizes the target parts across all experimental datasets. In all datasets, there exist parts that cannot be distinguished based on intensity alone; however, ParaView-MCP relies on a 1D TF, making such separations inherently challenging, which is reflected in its results across all datasets. In ParaView-MCP, the MLLM iteratively refines the TF by observing the rendered images and updating the intensity ranges corresponding to each part.

In the ParaView-MCP (text-only) setting, the MLLM first infers the intensity ranges of each part based on the given part names, and then begins updating the TF. In contrast, in ParaView-MCP (2D semantic), the MLLM leverages the provided MIP and segmentation mask to identify the intensity ranges of each part before initiating TF updates.

SAM3D successfully separates parts with relatively large and connected structures, but struggles to accurately segment thin or structurally complex parts with many components. This limitation aligns with observations reported by the SAM3D authors, who note that SAM3D, like its base model SAM, struggles with thin and branching structures. Furthermore, SAM3D requires users to generate point prompts by navigating slices along a chosen axis. Because parts often appear fragmented across slices, the annotations inevitably become discontinuous, leading to fragmented visualizations of the corresponding parts.

Bonsai Dataset. The parts are separated into blue leaves, green branches, red trunk, and yellow pot. In the ParaView-MCP results, the leaves are rendered semi-transparently by adjusting opacity, allowing the branches to be more clearly visible. However, parts with similar intensity values are not perfectly distinguished, such as the leaves and the ground beneath the pot, as well as the branches and the trunk. In ParaView-MCP (text-only), the trunk is visualized in green rather than red. In ParaView-MCP (2D semantic), the result deviates from our target configuration, producing green leaves and blue branches instead of blue leaves and green branches. The MLLM initially interprets the segmentation mask by associating the blue region with "inner foliage / dense branches" and the green region with "outer branches / lighter foliage," and begins updating the TF accordingly. In its final response, it settles on a configuration described as "green canopy, blue inner foliage." In contrast, our method directly leverages the semantic decomposition encoded in the segmentation mask, enabling consistent mapping between parts and colors from the mask to the final volume visualization. In the SAM3D results, the pot is well separated, but the remaining parts are segmented more thickly than in the raw volume, and portions of the leaves and branches that were not annotated are missing in the visualization.

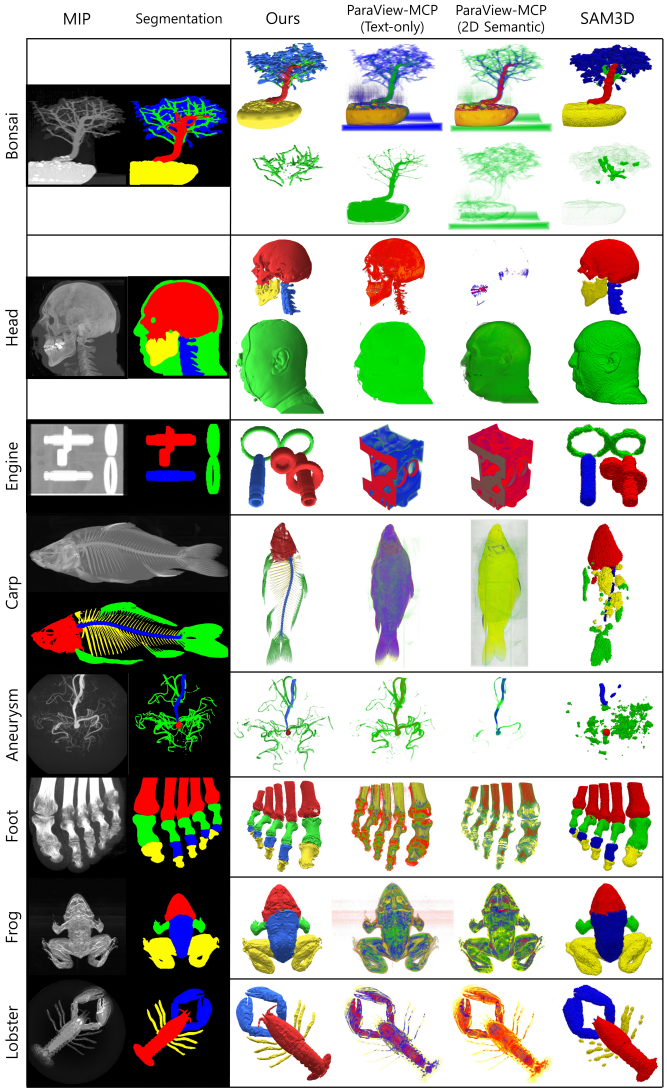


Figure 5: Comparison of semantic volume rendering results across eight datasets using different approaches. Each row corresponds to a dataset, showing the input MIP, the MLLM-generated segmentation mask, and the resulting visualizations produced by our method, ParaView-MCP under two configurations, and SAM3D.

Head Dataset. The parts are separated into red skull, yellow mandible, blue spine, and green skin. In the ParaView-MCP results, the skull, mandible, and spine are not clearly separated. In particular, ParaView-MCP (2D semantic) fails to properly visualize all three parts.

Table 1: Dataset statistics and execution time comparison across different methods.

					ParaView-MCP		SAM3D		
Dataset	**Resolution**	**Size (MB)**	**#Parts**	**Ours**	**Text-only**	**2D Semantic**	**User Prompt**	**Inference**	**Total**
Head	$128 \times 256 \times 256$	8	4	**43s**	2m 32s	2m 40s	26m 41s	21m 41s	48m 22s
Bonsai	$256 \times 256 \times 256$	16	4	**44s**	1m 29s	1m 13s	19m 23s	16m 10s	35m 33s
Lobster	$301 \times 324 \times 56$	5.2	3	**53s**	1m 37s	1m 54s	7m 24s	10m 35s	17m 59s
Carp	$256 \times 256 \times 512$	64	4	**1m 7s**	3m 19s	3m 3s	16m 1s	71m 55s	87m 56s
Engine	$256 \times 256 \times 128$	8	3	**49s**	1m 37s	2m 25s	8m 52s	6m 4s	14m 56s
Foot	$256 \times 256 \times 256$	16	4	**49s**	1m 30s	2m 18s	21m 35s	15m 38s	37m 13s
Frog	$256 \times 256 \times 44$	2.8	4	**47s**	2m 35s	2m 36s	11m 30s	11m 28s	22m 58s
Aneurysm	$512 \times 512 \times 512$	256	3	**56s**	2m 51s	2m 30s	10m 42s	23m 10s	33m 52s
			Average	**51s**	2m 11s	2m 20s	15m 16s	22m 5s	37m 21s

In the SAM3D results, the visualization appears clean, without the surrounding noise observed in both our method and ParaView-MCP. However, fine geometric details with complex curvature, such as the teeth and ears, are not separated as cleanly as in our method.

Engine Dataset. The parts are separated into red cylinders, blue crankshaft, and green connecting rods. In ParaView-MCP (text-only), not all parts are properly visualized; the MLLM confirms only the presence of the target colors and prematurely finalizes the TF. ParaView-MCP (2D semantic) also fails to correctly visualize all parts. In the SAM3D result, all parts are successfully distinguished; however, the connecting rods appear irregular and uneven compared to their actual structure.

Carp Dataset. The parts are separated into red skull, green fins, yellow ribs, and blue spine. In ParaView-MCP, the MLLM fails to correctly associate parts with their corresponding colors, misinterpreting the carp's skin as the spine or ribs and visualizing it instead. In the SAM3D result, the skull, which forms a relatively large connected structure, is well separated; however, the fins and spine appear fragmented, and the thin rib structures are difficult to separate accurately.

Aneurysm Dataset. The parts are separated into red aneurysm, blue parent vessel, and green vessels. In the results of all methods including ours, some vessels appear fragmented in the visualization. In the ParaView-MCP results, the vessels are separated from the other two parts; however, the aneurysm and parent vessel are not distinguished due to their similar intensity values. In the ParaView-MCP (2D semantic) setting, the intensity range is set relatively high to capture the brightly represented aneurysm in the MIP, which results in many vessels not being visualized. In the SAM3D result, the aneurysm, which forms a relatively large connected structure, is well separated; however, thin vessels are not successfully segmented, and the parent vessel is only partially and discontinuously visualized.

Foot Dataset. The parts are separated into red metatarsals, green proximal phalanges, blue middle phalanges, and yellow distal phalanges. Both our method and SAM3D produce results similar to the segmentation mask. In this dataset, the intensity ranges of the skin and bones partially overlap. As a result, in both our method and ParaView-MCP, portions of the bones become partially transparent during the process of removing the skin. In contrast, SAM3D produces cleaner boundaries between skin and bones; however, some voxels are misclassified, resulting in incorrectly labeled red and green regions.

Frog Dataset. The parts are separated into red head, blue torso, green arms, and yellow legs. Both our method and SAM3D produce results that show visual agreement with the segmentation mask. However, in the SAM3D result, the boundary between the head and torso is relatively less clean, as annotations for a single target part are performed across multiple slices. In the ParaView-MCP results, the parts cannot be separated based on intensity.

Lobster Dataset. The parts are separated into blue claws, red body, and yellow legs. In the ParaView-MCP results, the legs, which exhibit intensity differences, are separated and visualized in yellow; however, the claws and body, which cannot be distinguished based on intensity alone, appear with mixed colors. In the SAM3D result, the thin legs are fragmented and discontinuously visualized.

4.3.3 Execution Time Comparison

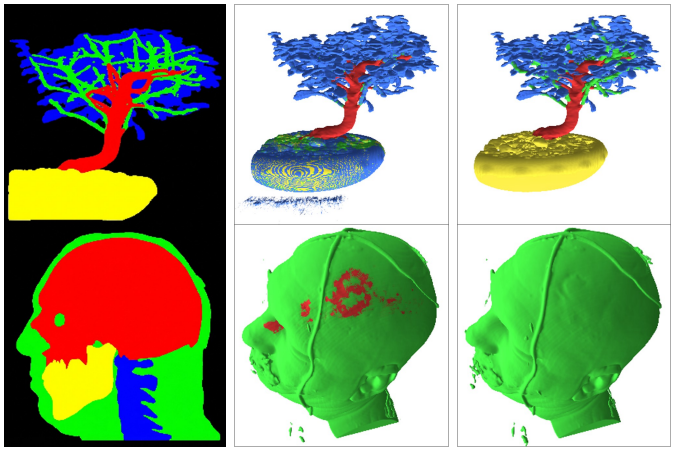
Table 1 shows the number of target parts and the time required to generate the comparison results shown in Fig. 5, for each method and dataset. For both our method and ParaView-MCP, the process proceeds automatically after a simple initial user input; therefore, we measure the total time from input entry to obtaining the final volume rendering image. In the case of SAM3D, we separately measure the user prompt creation time and the model inference time.

Across all datasets, not only on average but consistently, our method achieves the shortest execution time. This is because our approach requires minimal user input (volume name and MIP axis selection) and employs a lightweight algorithm and network, resulting in very fast computation. The most time-consuming step in our pipeline is the generation of the segmentation mask by the MLLM. ParaView-MCP also demonstrates short execution times. However, it requires an iterative process in which the rendered result is captured as a screenshot, interpreted by the MLLM, and used to update the TF repeatedly, leading to longer execution times compared to our method.

SAM3D requires significantly more time than the other two methods. Users must navigate through slices to identify target parts and provide point prompts. Since parts are often fragmented across slices, it is difficult to recognize them, and annotation becomes particularly challenging when structures are small or thin. For example, in the head dataset, the annotation area is large, and the skull appears thin and elongated in slices, making point prompt input time-consuming. In the foot dataset, identifying each bone based on its relative position is difficult in slice views, requiring repeated back-and-forth navigation across slices. Additionally, annotating numerous thin structures—such as leaves and branches in the bonsai dataset, and ribs in the carp dataset—increases the time. Moreover, since SAM3D processes one target part at a time, prompting and inference must be repeated for each part. The model inference time increases with the dataset size, the number of target parts, and the number of point prompts. Furthermore, SAM3D pads volumes to a 1:1:1 aspect ratio, which leads to increased processing time for some datasets.

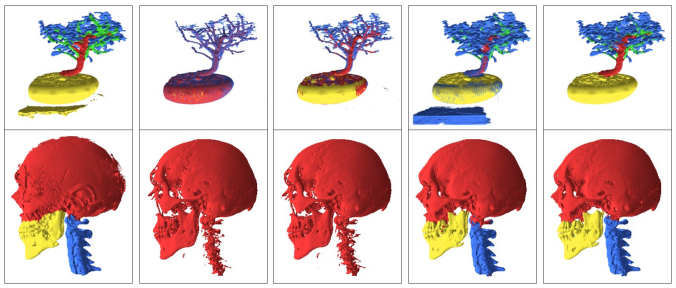
4.4 Ablation Study

Effect of Propagation Loss. Figure 6 compares the results obtained without and with the propagation loss $\mathscr{L}_{prop}$. Figure 6a shows the MLLM-generated segmentation mask used for the study. Without propagation loss (Fig. 6b), the results exhibit noticeable noise and inconsistency. For example, in the bonsai dataset, some voxels of the branches, the pot, and the ground in front of the pot are incorrectly classified. This occurs because, during the seed voxel construction process, only high-intensity voxels near the centers of the branches and the pot are selected as seed voxels. When trained using only the supervised loss on these seed voxels, some voxels in the bonsai dataset fail to receive semantic supervision from the seeds and are instead assigned to parts with more similar intensity values. In the head dataset, red noise appears on the skin surface. Since this area does not overlap with the



(a) Segmentation mask (b) w/o $\mathscr{L}_{\text{prop}}$ (c) w/ $\mathscr{L}_{\text{prop}}$

Figure 6: Comparison without and with propagation loss.



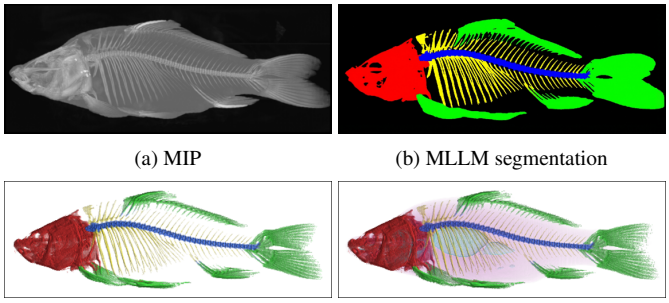
(a) $\mathbf{x}$ (b) $I(\mathbf{x})$ (c) $I(\mathbf{x})$,$G(\mathbf{x})$ (d) $\mathbf{x}$,$I(\mathbf{x})$ (e) All

Figure 7: Ablation study on input feature configurations for voxel-wise semantic prediction.

skin region in the segmentation mask, the corresponding voxels are not included as seed voxels. In this area, the skin and skull have similar intensity values and are spatially adjacent, causing some skin voxels to be misclassified as the red skull.

By introducing the propagation loss (Fig. 6c), the rendering results become significantly more spatially consistent. The propagation mechanism effectively transfers reliable information from seed voxels to neighboring regions based on feature similarity, suppressing isolated noisy predictions. As a result, both datasets show cleaner part boundaries and improved structural integrity. This demonstrates that the propagation loss plays a critical role in stabilizing the learning process and enhancing consistency in the proposed semantic volume rendering.

Effect of Input Feature Configurations. Figure 7 presents an ablation study on different input feature configurations for the voxel classification network. We use the masks shown in Figure 6a. When using only spatial coordinates $\mathbf{x}$ (Figure 7a), the leaves and branches in the bonsai dataset, which are spatially intertwined, are not well separated, and the ground in front of the pot is not clearly distinguished from the pot. In the head dataset, the ear region, which belongs to the skin, is misclassified as part of the skull. When using only intensity $I(\mathbf{x})$ (Figure 7b), semantic separation is barely achieved in both datasets. Incorporating both intensity and gradient magnitude $I(\mathbf{x}), G(\mathbf{x})$ (Figure 7c) further enhances boundary awareness, successfully separating the outer surface of the pot as a distinct yellow region in the bonsai dataset. When combining spatial and intensity features $\mathbf{x}, I(\mathbf{x})$ (Figure 7d), the results generally resemble the given segmentation mask, but still exhibit noticeable noise. Finally, using all inputs $\mathbf{x}, I(\mathbf{x}), G(\mathbf{x})$ (Figure 7e) produces the best performance, achieving the most accurate and consistent semantic volume rendering across both datasets. These results highlight that spatial and attribute features provide complementary information. Their combination is essential for achieving robust voxel-wise semantic prediction across different datasets in the proposed automatic TF design.



(a) MIP (b) MLLM segmentation

(c) Result of our method (d) Refined visualization

Figure 8: Limitation and refinement via post-processing.

5 Limitations and Future Work

Our method relies on the performance of the MLLM. In our implementation, we use *Gemini 3 Pro Image Preview (Nano Banana Pro)*, which produces high-quality segmentation masks, while other models often fail to achieve comparable quality. In addition, to enable automatic TF design with minimal user input, we rely on the MLLM to determine which parts to visualize; however, as a result, the parts that users intend to visualize are not always reflected in the generated segmentation. Another limitation arises from our reliance on MIP for semantic decomposition. Parts that are not visible in the MIP cannot be discovered and therefore cannot be visualized. For example, in the carp dataset, the air bladder is not visible in the MIP (Fig. 8a) due to its low intensity, and thus cannot be visualized using our method alone. Furthermore, when obtaining segmentation masks for datasets containing overlapping parts, such as the carp dataset, certain parts such as the skin are occasionally omitted, as shown in Fig. 8b. As a result, in our rendering output (Fig. 8c), both the skin and the air bladder are classified as background.

To address this limitation, a simple post-processing strategy can be applied. Among voxels classified as background, we first separate true background regions using a background mask automatically obtained by applying SAM or an MLLM to the MIP. Then, by performing k-means clustering with $k = 2$ on the remaining voxels based on intensity, we can recover and visualize the skin and air bladder, as shown in Fig. 8d. This demonstrates that our automatic TF design enables fast and easy visualization of major semantic parts, which can be further refined with minimal user input, such as selecting a small number of clusters.

For future work, we plan to explore extending the single-MIP-based seed construction to incorporate multiple projection views or MIPs decomposed into different depth-based layers, enabling multiple 2D semantic decompositions and reducing the risk of missing parts that are occluded or poorly represented in a single projection axis. We also aim to develop more flexible refinement strategies. For instance, visualizing voxel features using t-SNE could enable more fine-grained semantic separation. In addition, instead of fully relying on the MLLM, exploring methods that allow users to efficiently refine the MLLM-generated segmentation mask to better reflect their intent would lead to more accurate and controllable semantic volume rendering.

6 Conclusion

In this paper, we have presented a framework for automatic TF design via MLLM-assisted 2D semantic decomposition. By leveraging a single MIP image and minimal user input, our method transforms high-level semantic understanding from an MLLM into a dense voxel-wise representation, enabling direct construction of semantic TFs without iterative manual interaction. Experimental results demonstrate that our method produces semantically meaningful visualizations while significantly reducing the time and effort required for TF design compared to existing approaches. In particular, our framework enables intuitive part-level separation without relying on handcrafted feature spaces, repeated user interaction, or dataset-specific training. Overall, this work highlights the potential of integrating MLLM-based semantic understanding with volume rendering to simplify and enhance TF design.

Acknowledgments

This work was supported in part by the National Research Foundation of Korea under Grant RS-2024-00349697 and Grant RS-2021-NR060143; in part by the Institute for Information and Communications Technology Planning and Evaluation under Grant IITP-2026-RS-2020-II201819; in part by the Technology Development Program funded by the Ministry of SMEs and Startups (MSS) under Grant RS-2024-00437796; and in part by Korea University Grant.

References

[1] A. Abdelreheem, I. Skorokhodov, M. Ovsjanikov, and P. Wonka. Satr: Zero-shot semantic segmentation of 3d shapes. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 15166–15179, 2023. doi: 10.1109/ICCV51070.2023.01392 3

[2] J. Ahrens, B. Geveci, and C. Law. Paraview: An end-user tool for large data visualization. *The visualization handbook*, 717(8), 2005. 2

[3] K. Ai, K. Tang, and C. Wang. Nli4volvis: Natural language interaction for volume visualization via llm multi-agents and editable 3d gaussian splatting. *IEEE Transactions on Visualization and Computer Graphics*, 32(1):46–56, 2026. doi: 10.1109/TVCG.2025.3633888 2

[4] T. Brooks, A. Holynski, and A. A. Efros. Instructpix2pix: Learning to follow image editing instructions. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 18392–18402, 2023. doi: 10.1109/CVPR52729.2023.01764 2

[5] M. Caron, H. Touvron, I. Misra, H. Jegou, J. Mairal, P. Bojanowski et al. Emerging properties in self-supervised vision transformers. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 9630–9640, 2021. doi: 10.1109/ICCV48922.2021.00951 2

[6] T. J. Chan, A. Sahni, Y. Fang, J. Li, A. Luthra, A. Pouch et al. Sam3d: zero-shot semi-automatic segmentation in 3d medical images with the segment anything model. In *Medical Imaging 2025: Image Processing*, vol. 13406, pp. 574–580. SPIE, 2025. doi: 10.1117/12.3047242 3, 7

[7] W. Cho, D. Choi, H. Lim, J. Choi, S. Choi, H.-s. Min et al. Slice and conquer: a planar-to-3d framework for efficient interactive segmentation of volumetric images. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 7614–7623, 2024. doi: 10.1109/WACV57701.2024.00744 3

[8] S. Choi, H. Song, J. Kim, T. Kim, and H. Do. Click-gaussian: Interactive segmentation to any 3d gaussians. In *Computer Vision – ECCV 2024: 18th European Conference, Milan, Italy, September 29–October 4, 2024, Proceedings, Part III*, pp. 289–305. Springer-Verlag, Berlin, Heidelberg, 2024. doi: 10.1007/978-3-031-72646-0_17 2

[9] R. A. Drebin, L. Carpenter, and P. Hanrahan. Volume rendering. In *Proceedings of the 15th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '88, pp. 65–74. Association for Computing Machinery, New York, NY, USA, 1988. doi: 10.1145/54852.378484 1, 2

[10] Y. Du, F. Bai, T. Huang, and B. Zhao. Segvol: Universal and interactive volumetric medical image segmentation. *Advances in Neural Information Processing Systems*, 37:110746–110783, 2024. doi: 10.52202/079017-3516 3

[11] D. Engel, L. Sick, and T. Ropinski. Leveraging self-supervised vision transformers for segmentation-based transfer function design. *IEEE Transactions on Visualization and Computer Graphics*, 31(8):4357–4368, 2025. doi: 10.1109/TVCG.2024.3401755 1, 2

[12] Y. He, P. Guo, Y. Tang, A. Myronenko, V. Nath, Z. Xu et al. Vista3d: A unified segmentation foundation model for 3d medical imaging. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 20863–20873, 2025. doi: 10.1109/CVPR52734.2025.01943 2, 3

[13] F. Isensee, P. F. Jaeger, S. A. Kohl, J. Petersen, and K. H. Maier-Hein. nnu-net: a self-configuring method for deep learning-based biomedical image segmentation. *Nature methods*, 18(2):203–211, 2021. doi: 10.1038/s41592-020-01008-z 2

[14] S. Jeong, J. Li, C. R. Johnson, S. Liu, and M. Berger. Text-based transfer function design for semantic volume rendering. In *2024 IEEE Visualization and Visual Analytics (VIS)*, pp. 196–200. IEEE, 2024. doi: 10.1109/VIS55277.2024.00047 2

[15] B. Kerbl, G. Kopanas, T. Leimkühler, and G. Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics*, 42(4):1–14, 2023. doi: 10.1145/3592433 2

[16] C. M. Kim, M. Wu, J. Kerr, K. Goldberg, M. Tancik, and A. Kanazawa. Garfield: Group anything with radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 21530–21539, 2024. doi: 10.1109/CVPR52733.2024.02034 3

[17] S. Kim, Y. Jang, and S.-E. Kim. Image-based tf colorization with cnn for direct volume rendering. *IEEE Access*, 9:124281–124294, 2021. doi: 10.1109/ACCESS.2021.3100429 2

[18] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson et al. Segment anything. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 3992–4003, 2023. doi: 10.1109/ICCV51070.2023.00371 2

[19] J. Kniss, G. Kindlmann, and C. Hansen. Multidimensional transfer functions for interactive volume rendering. *IEEE Transactions on Visualization and Computer Graphics*, 8(3):270–285, 2002. doi: 10.1109/TVCG.2002.1021579 1, 2

[20] M. Levoy. Display of surfaces from volume data. *IEEE Computer graphics and Applications*, 8(3):29–37, 2002. doi: 10.1109/38.511 6

[21] L. Li, S. Lian, Z. Luo, B. Wang, and S. Li. Vclipseg: voxel-wise clip-enhanced model for semi-supervised medical image segmentation. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pp. 692–701. Springer, 2024. doi: 10.1007/978-3-031-72114-4_66 3

[22] M. Liu, M. A. Uy, D. Xiang, H. Su, S. Fidler, N. Sharp et al. Partfield: Learning 3d feature fields for part segmentation and beyond. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 9704–9715, 2025. doi: 10.1109/ICCV51701.2025.00905 3

[23] S. Liu, H. Miao, and P.-T. Bremer. Paraview-mcp: An autonomous visualization agent with direct tool use. In *2025 IEEE Visualization and Visual Analytics (VIS)*, pp. 61–65. IEEE, 2025. doi: 10.1109/VIS60296.2025.00018 2, 6

[24] J. Ma, Y. He, F. Li, L. Han, C. You, and B. Wang. Segment anything in medical images. *Nature Communications*, 15(1):654, 2024. doi: 10.1038/s41467-024-44824-z 2, 3

[25] J. Ma, Z. Yang, S. Kim, B. Chen, M. Baharoon, A. Fallahpour et al. Medsam2: Segment anything in 3d medical images and videos. *arXiv preprint arXiv:2504.03600*, 2025. doi: 10.48550/arXiv.2504.03600 2

[26] N. Max. Optical models for direct volume rendering. *IEEE Transactions on Visualization and Computer Graphics*, 1(2):99–108, 2002. doi: 10.1109/2945.468400 5

[27] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*, 65(1):99–106, 2021. doi: 10.1145/3503250 2

[28] B. Pan, J. Lu, H. Li, W. Chen, Y. Wang, M. Zhu et al. Differentiable design galleries: A differentiable approach to explore the design space of transfer functions. *IEEE Transactions on Visualization and Computer Graphics*, 30(1):1369–1379, 2023. doi: 10.1109/TVCG.2023.3327371 2

[29] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal et al. Learning transferable visual models from natural language supervision. In M. Meila and T. Zhang, eds., *Proceedings of the 38th International Conference on Machine Learning*, vol. 139 of *Proceedings of Machine Learning Research*, pp. 8748–8763. PMLR, 18–24 Jul 2021. 2

[30] N. Ravi, V. Gabeur, Y.-T. Hu, R. Hu, C. Ryali, T. Ma et al. Sam 2: Segment anything in images and videos. In *International Conference on Learning Representations*, vol. 2025, pp. 28085–28128, 2025. 3

[31] K. P. Soundararajan and T. Schultz. Learning probabilistic transfer functions: A comparative study of classifiers. In *Computer Graphics Forum*, vol. 34, pp. 111–120. Wiley Online Library, 2015. doi: 10.1111/cgf.12623 2

[32] K. Tang, K. Ai, J. Han, and C. Wang. Texgs-volvis: Expressive scene editing for volume visualization via textured gaussian splatting. *IEEE Transactions on Visualization and Computer Graphics*, 32(1):933–943, 2026. doi: 10.1109/TVCG.2025.3634643 2

[33] F.-Y. Tzeng, E. Lum, and K.-L. Ma. An intelligent system approach to higher-dimensional classification of volume data. *IEEE Transactions on Visualization and Computer Graphics*, 11(3):273–284, 2005. doi: 10.1109/TVCG.2005.38 2

[34] Y. Wang, B. Pan, K. Wang, H. Liu, J. Mao, Y. Liu et al. Intuitf: Mllm-guided transfer function optimization for direct volume rendering. *arXiv preprint arXiv:2506.18407*, 2025. doi: 10.48550/arXiv.2506.18407 2

[35] Y. Yang, Y. Huang, Y.-C. Guo, L. Lu, X. Wu, E. Y. Lam et al. Sampart3d: Segment any part in 3d objects. *arXiv preprint arXiv:2411.07184*, 2024. doi: 10.48550/arXiv.2411.07184 2, 3

[36] M. Ye, M. Danelljan, F. Yu, and L. Ke. Gaussian grouping: Segment and edit anything in 3d scenes. In *Computer Vision – ECCV 2024: 18th European Conference, Milan, Italy, September 29–October 4, 2024, Pro-*

ceedings, Part XXIX, pp. 162–179. Springer-Verlag, Berlin, Heidelberg, 2024. doi: 10.1007/978-3-031-73397-0_10 2, 3

[37] S. Yoo, S. Kim, and Y. Jang. Two-level transfer functions using t-sne for data segmentation in direct volume rendering. *IEEE Transactions on Visualization and Computer Graphics*, 31(9):5948–5961, 2025. doi: 10.1109/TVCG.2024.3484471 2

[38] H.-Y. Zhou, J. Guo, Y. Zhang, X. Han, L. Yu, L. Wang et al. nnformer: Volumetric medical image segmentation via a 3d transformer. *IEEE Transactions on Image Processing*, 32:4036–4045, 2023. doi: 10.1109/TIP.2023.3293771 2